Computer Organization and Design (4th)
by Hennessy, Patterson
Chapter 5.14, Problem 1E

Step 1

a) Given data:

Base CPI	Privileged O/S accesses per 10,000 instructions	Performance impact to trap to the guest O/S	Performance impact to trap to VMM	I/O accesses per 10,000 instructions	I/O access time(includes time to trap to guest O/S)
2	100	20cycles	150 cycles	20	1000 cycles

Let us assume no access to I/O.

CPI = BaseCPI + ((Priviliged o/s accesses per 10,000 instr) × (Performance impact to trap to the guest o/s+performance impact to trap to VMM))

$$CPI = 2 + ((100/10{,}000) \times (20+150))$$

$$= 2 + (0.01 \times 170)$$

$$= 2 + 1.7$$

$$= 3.7$$

$$\boxed{\therefore CPI = 3.7}$$

Step 2

VMM performance impact doubles:

CPI= BaseCPI + ((Priviliged o/s accesses per 10,000 instr) × (Performance impact to trap to the guest o/s+performance impact to trap to VMM))

$= 2+((100/10{,}000) \times (20+2\times 150))$

$= 2+(0.01\times 320)$

$= 2+3.2$

$= 5.2$

$\boxed{\therefore \text{CPI} = 5.2}$

Step 3

VMM performance impacts half:

CPI= BaseCPI + ((Priviliged o/s accesses per 10,000 instr) × (Performance impact to trap to the guest o/s+0.5 × performance impact to trap to VMM))

$\text{CPI} = 2+((100/10{,}000) \times (20+0.5\times 150))$

$= 2+(0.01\times 95)$

$= 2+0.95$

$= 2.95$

$\boxed{\therefore \text{CPI} = 2.95}$

Step 4

Obtain a 10% performance degradation, then

$1.1\times$(BaseCPI + (Priviliged o/s accesses per 10,000 instr $\times$ Performance impact to trap to the guest o/s)) = BaseCPI + ((Priviliged o/s accesses per 10,000 instr) $\times$ (Performance impact to trap to the guest o/s + n))

$$1.1\times(2+(100/10,000)\times 20) = 2+(100/10,000\times(20+n))$$

$$2.42 = 2+ (0.01 * (20+n)$$

n = 22 cycles is longest possible penalty to trap to the VMM.

Step 5

b) Given data:

Base CPI	Privileged O/S accesses per 10,000 instructions	Performance impact to trap to the guest O/S	Performance impact to trap to VMM	I/O accesses per 10,000 instructions	I/O access time(includes time to trap to guest O/S)
1.5	110	25cycles	160 cycles	10	1000 cycles

Let us assume no access to I/O.

CPI = BaseCPI + ((Priviliged o/s accesses per 10,000 instr) × (Performance impact to trap to the guest o/s+performance impact to trap to VMM))

$$CPI = 1.5 + ((110/10{,}000) \times (25+160))$$

$$= 1.5 + (0.011 \times 185)$$

$$= 1.5 + 2.035$$

$$= 3.535$$

$$\boxed{\therefore CPI = 3.535}$$

Step 6

VMM performance impact doubles:

CPI = BaseCPI + ((Priviliged o/s accesses per 10,000 instr) × (Performance impact to trap to the guest o/s+performance impact to trap to VMM))

$$CPI = 1.5 + ((110/10{,}000) \times (25 + 2 \times 160))$$

$$= 1.5 + (0.011 \times 345)$$

$$= 1.5 + 3.795$$

$$= 5.295$$

$$\boxed{\therefore CPI = 5.295}$$

Step 7

VMM performance impacts half:

CPI = BaseCPI + ((Priviliged o/s accesses per 10,000 instr) × (Performance impact to trap to the guest o/s+0.5 × performance impact to trap to VMM))

$$CPI = 1.5 + ((110/10{,}000) \times (25 + 0.5 \times 160))$$

$$= 1.5 + (0.011 \times 105)$$

$$= 1.5 + 1.155$$

$$= 2.655$$

$$\boxed{\therefore CPI = 2.655}$$

Step 8

Obtain 10% performance degradation, then

$$1.1\times(\text{BaseCPI} + (\text{Priviliged o/s accesses per 10,000 instr} \times \text{Performance impact to trap to the guest o/s})) = \text{BaseCPI} + ((\text{Priviliged o/s accesses per 10,000 instr}) \times (\text{Performance impact to trap to the guest o/s} + n))$$

$$1.1\times(1.5+(110/10,000)\times 25) = 1.5+(110/10,000\times(25+n))$$

$$1.9525 = 1.5+0.275+0.011 * n$$

$n = 16.13$ cycles is longest possible penalty to trap to the VMM.

Step 1

a) Given data:

Base CPI = 2

Privileged OS access = 100 per 10,000 instructions

Performance impact to trap to the guest O/S = 20 cycles

Performance impact to trap to VMM = 150 cycles

I/O accesses = 20 per 10,000 instructions

I/O access time = 1000 cycles

Step 2

$CPI_{Non\ virtualised} = CPI_{base}$ +(prev OS access-I/O accesses)/10000 × prev impact trapguest OS+I/O access/10000 × I/O access time.

$CPI_{non\ virtualized} = 2.0 + (100 - 20)/10{,}000 \times 20 + 20/10{,}000 \times 1000$

$= 2+80/10{,}000\times20+ 20/10{,}000 \times 1000$

$= 2.0 + 0.16 + 2.0$

$= 4.16$

Step 3

$CPI_{Virtulaised} = CPI_{base} + (\text{prev OS accesses})/10000 \times (\text{perf impact trap guest OS} + \text{pref impact trap VMM}) + (\text{I/O accesses}/10000) \times (\text{I/O access time} + \text{pref impact trap VMM})$

$CPI_{virtualized} = 2.0 + 80/10{,}000 \times (20 + 150) + 20/10{,}000 \times (1000 + 150)$

$= 2.0 + 80/1000 \times 170 + 20/10000 \times 1150$

$= 2.0 + 1.36 + 2.3$

$= 5.66$

I/O bound applications have a smaller impact from virtualization because, comparatively, a much longer time is spent on waiting for the I/O accesses to complete.

Step 4

b) Given data:

Base CPI = 1.5

Privileged OS access = 110 per 10,000 instructions

Performance impact to trap to the guest O/S = 25 cycles

Performance impact to trap to VMM = 160 cycles

I/O accesses = 10 per 10,000 instructions

I/O access time = 1000 cycles

Step 5

$CPI_{\text{Non virtualised}} = CPI_{\text{base}}$+(prev OS access-I/O accesses)/10000 × prev impact trapguest OS)+I/O access/10000 × I/O access time.

$CPI_{\text{non virualised}} = 1.5 + (110-10) / 10000 \times 25 + 10 / 10000 \times 1000$

$= 1.5 + 100/10000 \times 25 + 10/10000 \times 1000$

$= 1.5 + 0.25 + 1$

$= 2.75$

Step 6

$CPI_{\text{Virulaised}} = CPI_{\text{base}}$+(prev OS accesses)/10000 × (perf impact Trap guest OS+ pref impact trap VMM)+(I/O accesses/10000) × (I/O access time +pref impact trap VMM)

$CPI_{\text{virtualised}} = 1.5 + (110-10)/10000 \times (25+160) + 10/10000 \times (1000+160)$

$= 1.5 + 100/10000 \times 185 + 10/10000 \times 1160$

$= 1.5 + 1.85 + 1.16$

$= 4.51.$

I/O bound applications have a smaller impact from virtualization because, comparatively, a much longer time is spent on waiting for the I/O accesses to complete.

Computer Organization and Design (4th)
by Hennessy, Patterson
Chapter 5.14, Problem 3E

Step 1

Virtual memory aims to provide each application with the illusion of the entire address space of the machine whereas Virtual machines aims to provide each operating system with the illusion of having the entire machine to its disposal. Thus they both serve very similar goals, and offer benefits such as increased security. Virtual memory can allow for many applications running in the same memory space to not have to manage keeping their memory separate.

Computer Organization and Design (4th)
by Hennessy, Patterson
Chapter 5.14, Problem 4E

Step 1

Emulating a different ISA requires specific handling of that ISA's API. Each ISA has specific behaviors that will happen upon instruction execution, interrupts, trapping to kernel mode, etc. that therefore must be emulated. This can require many more instructions to be executed to emulate each instruction than was originally necessary in the target ISA. This can cause a large performance impact and make it difficult to properly communicate with external devices. An emulated system can potentially run faster than on its native ISA if the emulated code can be dynamically examined and optimized.

Step 2

For example, if the underlying machine's ISA has a single instruction that can handle the execution of several of the emulated system's instructions, then potentially the number of instructions executed can be reduced. This is similar to the recent Intel processors that do micro- op fusion, allowing several instructions to be handled by fewer instructions.